AMENDMENTS TO THE CLAIMS

1. (Original) A method of optimizing computer program code where the computer program code includes a plurality of statements, the method comprising the steps of:

identifying a keyword statement;

searching the program code for the keyword statement;

determining if the keyword statement begins a repeating pattern of statements in the program code; and

replacing the repeating pattern of statements with a program loop equivalent to the repeating pattern of statements.

2. (Original) The method of optimizing as set forth in claim 1 wherein the keyword statement includes a keyword and an optional data reference, the method further including prior to the searching step:

sequentially locating each keyword statement in the program code; and

converting the optional data reference, if present, from each located keyword statement to a data array reference.

3. (Original) The method of optimizing as set forth in claim 2 wherein the converting includes assigning an array index value to the data array reference where each located keyword statement is assigned a next sequential value of the array index value.

4. (Original) The method of optimizing as set forth in claim 3 wherein the determining step further includes:

comparing data array references of two keyword statements from the program code; and

determining if the array index values from the data array references match in size and sequential order.

5. (Original) The method of optimizing as set forth in claim 1 wherein the determining step includes:

determining a first pattern of statements in the program code beginning with a first keyword statement and ending with a statement preceding a second keyword statement that sequentially appears in the program code after the first keyword statement;

determining a second pattern of statements in the program code beginning with the second keyword statement and ending with a statement preceding a third keyword statement that sequentially appears in the program code after the second keyword statement; and

comparing the first pattern of statements to the second pattern of statements; and

setting the first pattern of statements as a repeating pattern if the first and second pattern of statements substantially match.

6. (Original) The method of optimizing as set forth in claim 1 wherein the replacing step includes:

generating loop code for executing a loop within the source code at a location of the repeating pattern of statements;

inserting one instance of the repeating pattern of statements within the loop code; and

defining the loop code to iterate a number of times equal to a number of instances of the repeating pattern.

7. (Original) The method of optimizing as set forth in claim 1 wherein the keyword statement is identified from a predetermined keyword statement.

8. (Original) The method of optimizing as set forth in claim 1 wherein the keyword statement is identified from a selection made by a user.

9. (Original) The method of optimizing as set forth in claim 1 further including identifying a plurality of keyword statements and repeating the method for optimizing for each of the plurality of keyword statements.

10. (Original) A software code optimizer comprising:

analyzing program instructions for analyzing a software code and determining an occurrence of a repeating pattern of code therein; and

converting program instructions for converting the repeating pattern of code to a programming loop that performs an equivalent function as the repeating pattern of code.

11. (Original) The software code optimizer as set forth in claim 10 wherein the analyzing program instructions further include:

program instructions for searching the software code for a keyword; and

program instructions for identifying if the keyword begins a repeating pattern of code within the software code and determining a number of occurrences of the repeating pattern.

12. (Original) The software code optimizer as set forth in claim 11 further including program instructions for repeating the analyzing program instructions for a plurality of keywords.

13. (Original) The software code optimizer as set forth in claim 11 wherein the converting program instructions further include program instructions for setting the programming loop to repeat a number of times equal to the number of occurrences of the repeating pattern and inserting one occurrence of the repeating pattern within the programming loop.

14. (Original) The software code optimizer as set forth in claim 10 further including program instructions for locating data references in each statement of program code containing the keyword and converting the data references to data array references.

15. (Original) The software code optimizer as set forth in claim 10 further including a compiler for translating the software code to an object code executable by a computer.

16. (Original) A process for optimizing a software code that includes a plurality of statements, the process comprising the steps of:

locating multiple occurrences of a code pattern within the software code where the multiple occurrences appear sequentially to each other in the software code;

generating a program loop that executes one occurrence of the code pattern a number of times to produce an equivalent result as executing the multiple occurrences of the code pattern; and

replacing the multiple occurrences of the code pattern in the software code with the program loop.

17. (Original) The process for optimizing a software code as set forth in claim 16 further including:

selecting a keyword statement; and

defining the code pattern based on the keyword statement.

18. (Original) The process for optimizing a software code as set forth in claim 17 wherein the defining step includes:

locating a first instance of the keyword statement in the software code;

defining a first code pattern to include at least the first instance of the keyword statement;

adding subsequent non-keyword statements to the first code pattern until a second instance of the keyword statement appears in the software code;

defining a second code pattern to include at least the second instance of the keyword statement;

adding subsequent non-keyword statements to the second code pattern until a third instance of the keyword statement appears in the software code or until a number of the subsequent non-keyword statements added equal a number of the subsequent non-keyword statements in the first code pattern; and

comparing the first code pattern with the second code pattern to determine if the second code pattern is a multiple occurrence of the first code pattern.

19. (Original) The process for optimizing a software code as set forth in claim 16, prior to the locating step, further including:

selecting at least one keyword statement where the keyword statement includes a keyword and an optional data reference; and

converting each of the data references that appear in each keyword statement in the software code to a data array reference, the data array reference being loaded with values of the converted data references.

20. (Original) The process for optimizing a software code as set forth in claim 16 wherein the generating a program loop step includes generating a looping instruction.

21. (New) A method of optimizing computer program source code, wherein the computer program source code includes a plurality of statements, comprising:

searching the program source code for a keyword statement;

determining whether the keyword statement begins a first pattern of one or more statements in the program code that is repeated in a second set of statements in the program source code; and

replacing the first pattern of one or more statements and the second set of statements in the program source code with a loop control statement and at least one loop-body statement that together define a function equivalent to the first pattern and second set of statements.

22. (New) The method of claim 21, further comprising, wherein the keyword statement includes a keyword and an associated data value, and prior to the determining step, for each keyword statement in the program code, replacing in the program source code each data value with a selected array name and an associated array index specification sized to store the associated data value.

23. (New) The method of claim 22, wherein the replacing step includes specifying in each array index specification a non-overlapping range of array index values.

24. (New) The method of claim 23, wherein the specifying step includes specifying sequential ranges of array index values in array index specifications associated with keyword statements that occur sequentially in the program source code.

25. (New) The method of claim 24, further comprising, wherein the replacing step includes generating the loop control statement and the at least one loop body statement to reference the array name and each sequential, non-overlapping range of index values.

26. (New) An apparatus for optimizing computer program source code, wherein the computer program source code includes a plurality of statements, comprising:

means for searching the program source code for a keyword statement;

means for determining whether the keyword statement begins a first pattern of one or more statements in the program code that is repeated in a second set of statements in the program source code; and

means for replacing the first pattern of one or more statements and the second set of statements in the program source code with a loop control statement and at least one loop-body statement that together define a function equivalent to the first pattern and second set of statements.

27. (New) An apparatus for optimizing computer program code where the computer program code includes a plurality of statements, comprising:

means for identifying a keyword statement;

means for searching the program code for the keyword statement;

means for determining if the keyword statement begins a repeating pattern of statements in the program code; and

means for replacing the repeating pattern of statements with a program loop equivalent to the repeating pattern of statements.